Andrew Gibson

DS 210 Final Project Writeup

Collaborated with:

None

Sources:

Stackoverflow

Lectures 25, 26, 27, 28, 30, 31, 34, 35, 36

Rust language website

workflow:

I followed my proposal pretty much exactly, however I ended up doing more work near the final dude date due to a family emergency, so I couldn’t do anything the Friday and Saturday before the final due date. I kept the same data set, but I ended up adjusting what I wanted to analyze. I first wanted to focus on centrality, but I shifted to average distances because I was not sure what specific aspect of centrality I wanted to focus on AND average distances seemed like it would help draw roughly the same conclusions as centrality, which was “central” (or close) California’s road network is. HOWEVER, I added a measure of clustering to see which nodes were closest together. When writing the code, I started with data validation. My dataset was simple, so it did not need any adjustments besides ignoring the first 4 lines of the txt file. Each line was just two numbers separated by white space which was easy to tell the code how to read. I then moved to how the code reads the data, by assigning the nodes to variables. Then I used those variables to calculate my results with my next two modules. I then put it all together in my main.rs file. Once I was satisfied, I realized I needed more complexity, so I decided to add clustering calculation to stay true to my former goal. Finally, I ran tests with a very simple list of nodes to verify everything worked. Most errors I ran into involved variable types. I had to ensure that my variables followed a cohesive set of variable types based on the numbers I was getting involved with. I regularly committed my changes, usually after getting one module of my code working, when I ran the program, printing, and for small adjustments.

what the project does:

My DS 210 final project analyzes a dataset of the nodes representing California’s road network. Each node represents an intersection with another node. In my project, I used rust to calculate the average distances and clustering coefficients between all other nodes based on a random sample. This is calculated by 1) gathering a random sample of pairs of nodes 2) Calculating the average distance of each unique node in the sample based on all the nodes it shares with. To start, I separated my project into 5 different rust files. These files are: read\_and\_parse.rs, utilize.rs, analysis.rs, main.rs, and tests.rs. The first three rust files are modules used in my main and tests rust files. I will be explaining each file in order and how the code works. To start, the read\_and\_parse.rs file uses several methods to read files and save as a Hashmap. It starts by reading the file line by line, where each line represents an edge in a graph. It then parses each line into two nodes, then constructs a list. It uses types HashMap<i32> and Vec<i32>. It also contains print statements which says an error has occurred, which helped on occasion. Next, the utilize.rs file focuses on utilizing BFS. In this case, breadth first search is BFS is used to calculate distances from a given starting node to other nodes in the graph. The function returns a mapping of distances from the start node to all its other nodes. This is particularly helpful for average distances because I need to keep track of which nodes is the “start” in each case. Next, I worked on the analysis.rs file. This file is where I calculated everything. First, the average\_distance function takes the previous BFS function to calculate all the distances then averages them out. Second, the random\_sample function takes a random sample of edges from my list of nodes and returns it. Then, the unique\_nodes function tells me how many unique nodes I have in that said sample. This uses a HashSet to keep track of nodes. It iterates through, and if the list already contains that node, it does not add it. Next, I put it all together in my main.rs file. I added my previous modules with methods from each. The main function first sets variables filename and start\_time. The filename is set to my txt file I am analyzing and start\_time is set to 0. I use this to see how much time has passed when running. This function was fairly simple to set up. I separated it to separate sections which follow the overall workflow of the project. It prints the progress and time it has taken with all the results. More details on the structure of the code is found in comments.

what the output looks like:

Starting the program...

Reading and parsing the file...

Data parsed successfully.

Time taken to parse data: 2.26s

Getting a random sample...

Sample obtained.

Time taken to get sample: 2.83s

Calculating average distance...

Average distance calculated.

Time taken to calculate average distances: 3725.95s

Calculating clustering coefficient...

Clustering coefficient calculated.

Time taken to calculate clustering coefficient: 3729.18s

Sample size: 1000

First 20 lines of the random sample:

[(1861859, 1861857), (431484, 431485), (1605635, 1605636), (271426, 271421), (1071641, 1071639), (824148, 824130), (588679, 629448), (816704, 816706), (1911090, 1911092), (633016, 633036), (1743193, 1743191), (259684, 259779), (306123, 305897), (1224352, 1224353), (1484485, 1484487), (466645, 466628), (94318, 94310), (1783800, 1783801), (1746662, 1746660), (952171, 952199)]

Number of unique nodes in sample: 1999

Average distance between nodes: 311.50

Average clustering coefficient: 0.0376

Program finished.

This is what the output of my program looks like. Note, the exact values will vary based on the sample size, but after multiple runs I netted very similar results.

what I found:

After running this project I found a lot about CA’s road network and how rust runs. The average distance between node pairs was 311.5. Based on the context of this dataset, this suggests an overview higher average distance between nodes than expected, but very connected overall. California’s road network is fairly connected, but geographic features separate connections. In cities and/or flat areas there is a lower average distance, with rural and/or hilly areas having a less connections and therefore a higher average distance. The dense areas therefore draw the distance lower. Therefore, CA’s road network has a higher average compared to other states.

how to run it:

To run, make sure the working directory is set to the location of the files. The txt file should be in the same location. To adjust the sample size, navigate to the main.rs file and refer to the comment. Then simply do cargo run in the terminal and the results will gradually print. The distance calculations run time is scaled based on the size of the sample.